Diffuse Curvature Computation for Surface Recognition

J. M. Savignat, O. Stab, A. Rassineux, and P. Villon

Abstract. Diffuse approximation is a local approximation scheme based on a moving least square fit. Derivatives are estimated by a pseudo-derivation operator which (under certain conditions) converges towards the function derivatives. For this reason, we use it to compute curvature over triangular surfaces as an extention of the fitting algorithm. We also take triangle normals into account, which leads to a high quality curvature estimator. We develop a surface recognition algorithm for triangular surfaces based on this curvature computation on the one hand, and on the topology described by the mesh on the other hand. Its application allows us to treat successfully some real CAD models, implying that diffuse approximation is a powerful tool for surface modelling, and for derivative-based computations.

§1. Diffuse Approximation

We shall focus in this part on the 1D case because the extension to higher dimensions only involves notational difficulties. Given a set of points $(x_i)_{i\in I}$ in $\Omega \subset \mathbb{R}$ an open interval, with measures $(u_i)_{i\in I}$, we build locally an approximation of the underlying function u via an estimation of the Taylor expansion of the function u. It should be noted that for any function $u \in C^{m+1}$, the Taylor expansion of order m exists at each point y,

$$u(x) = \sum_{k=0}^{m} u^{(k)}(y)\frac{(x-y)^k}{k!} + \int_y^x \frac{(t-y)^m}{(m)!} u^{(m+1)}(t)\, dt,$$

and that the polynomial part is an approximation of u near the point y.

The estimate uses some weight functions w_i associated with each point x_i and locally supported around x_i. We define $I(x) = \{i \in I, w_i(x) \neq 0\}$ as the set of indices of data points whose weight function is non-null at x. The computation procedes by minimisation at a point y of the functional

$$\mathcal{E}_y(\{\alpha\}) = \sum_{i\in I} w_i(y)(\bar{u}_y(x_i) - u_i)^2,$$

Curve and Surface Design: Saint-Malo 1999
Pierre-Jean Laurent, Paul Sablonnière, and Larry L. Schumaker (eds.), pp. 363–370.

ISBN 0-8265-1356-5.

with $\bar{u}_y(x) = \sum\limits_{k=0}^{m} \alpha_k(y) p_y^k(x)$ and $\langle p_y(x)\rangle = \langle 1, (x-y), \ldots, \frac{(x-y)^k}{k!}, \ldots\rangle$. The approximation $\tilde{u}$ of u and its derivatives are the coefficients α_k: $\tilde{u}(y) = \alpha_0(y)$, $\frac{\delta \tilde{u}}{\delta x}(y) = \alpha_1(y)$, ... This approximation method was first proposed in [10], and efficient computation was dicussed in [2].

The diffuse approximation properties depend mainly on the weight functions w_i. Their usual form is $w_i(x) = w_{ref}(\frac{x-x_i}{\rho_i})$, where w_{ref} is a reference bell function with support $(-1,1)$, and ρ_i is the influence radius of point x_i. We shall suppose that these radii are chosen so that the approximation exists at any point x (i.e. $\forall x \in \Omega, \mathrm{Card}(I(x)) \geq m$). [3] and [4] presented a few techniques to calculate such radii. With these definitions, the main properties of $\tilde{u}$ are the following:

- $\tilde{u}$ has the same smoothness as w_{ref} (e.g. if $w_{ref} \in C^2$, $\tilde{u} \in C^2$).
- The approximation reproduces polynomial functions up to degree m.
- $\tilde{u}$ and the pseudo-derivatives $\frac{\delta^k \tilde{u}}{\delta x^k}$ $(k \leq m)$ converge to u and its derivatives when the number of data points increases (see [16]).
- The diffuse approximation is linear, and the shape functions defined by $\tilde{u}(x) = \sum\limits_{i \in I(x)} N_i(x) u_i$ are local, $\mathrm{supp}(N_i) = \mathrm{supp}(w_i)$.

§2. Hermite Approximation Scheme

We propose to also take differential data into account in the criterion $\mathcal{E}_y$ to build a Hermite approximation scheme. Let $(x_j)_{j \in J}$ denote the set of points at which some differential data $v_j = \mathcal{D}_j(u)(x_j)$ are known. We associate a weight function w_j with each point x_j. The modified criterion is

$$\mathcal{E}_y(\{\alpha\}) = \sum_{i \in I(y)} w_i(y)(\bar{u}_y(x_i) - u_i) + \sum_{j \in J(y)} \lambda_j w_j(y) ||\mathcal{D}_j \bar{u}_y(x_j) - v_j||^2.$$

It is not restrictive to suppose that all the differential operators correspond to the same operator $\mathcal{D} = \{\mathcal{D}^l\}_{l \in [1,n]}$. Then the vector $\{\alpha\}$ is a solution of the system

$$A(y)\{\alpha(y)\} = \{b(y)\},$$

with

$$A(y) = P^\top(y) W(y) P(y) + \lambda \sum_{l=1}^{n} P^{l\top}(y) W^d(y) P^l(y),$$

$$\{b(y)\} = P^\top W(y) U(y) + \lambda \sum_{l=1}^{n} P^{l\top}(y) W^d(y) V^l(y),$$

where $W(y)$ and $W^d(y)$ are the diagonal matrix of weights $w_i(y)$ and $w_j(y)$ respecively, $P(y) = [p_y(x_i)]_{i \in I(y)}$,$P^l(y) = [\mathcal{D}^l(p_y)(x_j)]_{j \in I(y)}$ and $U(y) = \{u_i\}_{i \in I(y)}$, $V(y) = \{v_j\}_{j \in J(y)}$.

The previous properties remain valid for this new formulation. A similar approximation method was proposed in [7] for dealing with boundary conditions in a Galerkin method for partial differential equation.

§3. Curvature Computation

Curvature is mainly used in the treatment of range images (see [1]), and most algorithms were developed for these kind of data. In the paper [9], the authors distinguished four types of algorithms: finite difference methods, the facet model, geometrical methods, and fitting methods. The first two categories only apply to depth maps, whereas the last two are more general.

As the geometrical methods are ad hoc contructions, we shall not examine them in this paper. The facet model described in [5] uses a polynomial fit to compute more accurate finite difference formulas. The same idea was used in [8] for generalized finite differences. Therefore, the last three methods are mainly of the same kind, and a diffuse model gives some theoretical background to them.

Except for geometrical methods, curvature computation at nodes (i.e. data points) is composed of four steps:

1) Extraction of the node neighborhood,
2) Calculation of a coordinate system in which the surface is a Monge patch (i.e. there exists a function φ such that the surface has the form $(x, y, z = \varphi(x, y))$),
3) Evaluation of partial derivatives of the surface at the node,
4) Computation of curvatures.

The finite difference and facet model-based methods precompute some steps to obtain faster estimates. A more extensive bibliography can be found in [9] and [13].

Meshed surface curvature estimation can be done in one of the following three ways:

1) Forget the mesh, and treat a 3D point set,
2) Use the mesh as a purely topological attribute,
3) Use the mesh to interpolate the data.

The paper [15] uses the second strategy: it applies a multiresolution fitting method where the neighborhood of a node is defined through the triangular mesh. " Different layers of connectivities define different levels of neighboring relationships, e.g., the first level neighbors are the point with direct connection with the node, the second level ones have direct connection with the first level neighbors, and so on ". The third solution is difficult because it needs high continuity elements which are difficult to build; but [12] shows a solution based on G^2 continuity which is not robust to element shape [13]. Our method is of the second kind.

§4. Diffuse Curvature Computation

The diffuse curvature computation uses both the point positions and some normals (e.g. triangle normals in this paper). We focus on the computation at the nodes only, which will help in the definition of the parameters of the diffuse algorithm. The polynomial basis is $\langle 1, x, y, x^2, xy, y^2 \rangle$ which allows the

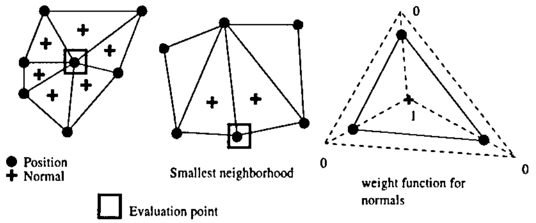

Fig. 1. Neighborhood selection.

evaluation of second-order partial derivatives needed for curvature computation.

We shall follow the four steps described above to compute curvature at a node x_i. Neighborhood extraction is based on the following weight functions: w_i is such that $w_i(x_i) = 1$, $w_i(x_{i'}) = \frac{1}{2}$ if node $x_{i'}$ is connected to x_i and $w_i(x_{i'}) = 0$ otherwise. The weight function of a normal is defined with the dashed triangulation (Figure 1, right). For smoothness reasons, we add the gray nodes to evaluate curvature at nodes on the border with only three edges (Figure 1, center).

We then compute local coordinates using an algorithm based on principal component analysis, and estimate the partial derivatives of the Monge patch (i.e. of function φ) defined by the data points with the pseudo-derivatives of the diffuse approximation at point x_i. Finally, principal curvatures k_1 and k_2 are computed with their associated directions.

A numerical study of the proposed method is given in [13], and shows that it gives at least as good results as the fitting method with smaller dependence neighborhoods, which is an important factor for surface recognition. It shows, moreover, that λ has to be small.

§5. Surface Recognition

Surface recognition is the first step in reconstructing a CAD model from a triangular mesh. We shall suppose in the following that the considered surface satifies the following hypotheses:

H$_0$: The surface is composed of parts of planes, cylinders, spheres, cones and torii (called patches).

H$_1$: Each patch contains at least one interior node.

H$_2$: Patch intersection are contained in the mesh (i.e. they are described by some edges chains).

Under these hypotheses, the above-mentioned diffuse curvature computation always estimates the real surface curvature at nodes interior to a patch (H$_1$ and H$_2$), because data are taken from the right surface. This is not the case with usual techniques (mainly the fitting method). This property is essential to proving that the recognition algorithm correctly classifies each node of a surface under the hypotheses H$_0$, H$_1$ and H$_2$ [14].

The recognition algorithm is composed of four steps.

Firstly, an initial classification (based on hypothesis H_0) is proposed with the following rules applied sequentially:

- If $k_1 = k_2 = 0$, the nodes is a PLANE node.
- If $k_2 \neq 0$ and $\frac{k_1}{k_2} = 1$, the node is a SPHERE node.
- If $k_2 \neq 0$ and $\frac{k_1}{k_2} \neq 1$, the node is a TORUS node.

The classification of non-classified nodes uses their comparison with connected nodes:

- If all connected nodes have the same k_1 and associated direction, the node is a CYLINDER node.
- If they have same k_1 and different directions, the node is a TORUS node.
- The node is a CONE node otherwise.

Secondly, we check the consistency of the initial classification with hypotheses H_0, H_1 and H_2. For example, a cone-cylinder intersection node is classified as a TORUS node. The basic idea of this consistency check is that classified connected nodes must form some connected homogeneous sets (as a consequence of H_2). From the study of intersections between the five primitives, it is possible to define three consistency rules (that are shown to be sufficient in [14]):

- For all connected nodes of different kinds: if one of them is a PLANE node unclassify the other one, otherwise if one of them is a CYLINDER node, unclassify the other one. Unclassify both nodes in other cases.
- If two connected nodes are both TORUS nodes with different k_1, unclassify both nodes.
- If two connected nodes are both SPHERE nodes with different k_2, unclassify both nodes.

At this stage, we obtain some germs that are homogeneous connected sets of nodes. We shall grow these germs to classify the whole surface via a marching algorithm.

Thirdly, we consider a classified node $\boldsymbol{n}$ and the patch P to which it belongs. From hypothesis H_2, for any triangle $T = (\boldsymbol{n}, m, p)$ we can claim that

1) T and its edges $\boldsymbol{n}m$ and $\boldsymbol{n}p$ belong to the interior of P,

2) Nodes m, p and edge mp belong to P.

Therefore, nodes m and p are either in the interior of the patch P or on the intersection of P and another patch. This analysis forms the topological operator of the marching algorithm.

The next step is then to check whether nodes m and p are interior nodes or intersection nodes. This decision is based on two rules which use geometrical

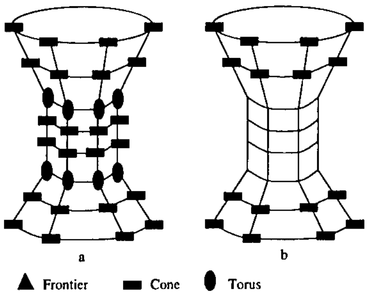

Fig. 2. (a) Initial classification (b) Consistent classification.

information. The first rule concerns connected triangles and their connecting edge.

- If one of the triangles is not classified, do nothing,.
- If both triangles are of the same kind and a vertex of the common edge belongs to an intersection, classify the other vertex as an intersection node.
- If the triangles are of a different kind, the vertices of the common edge are intersection nodes.

The second rule is node-based. It looks at the classification of the connected nodes: If this list is not homogeneous, then the node lies on an intersection. If it is homogeneous, some tests based on the same ideas as the initial classification allow us to check whether the node belongs to a surface or to an intersection of two surfaces of the same kind (this situation may happen after some iterations of the marching algorithm). The iteration of the topological operator and the two classification rules grows the germs of the consistent initial classification.

Figure 2 shows that the initial consistency algorithm may kill all potential germs of some patches. Provided that hypothesis H_1 is valid, some post treatment can be applied to this situation. The basic ideas of these treatments are the same as those being used in the main recognition algorithm. This is the fourth and last step of the classification.

§6. Conclusion

Under the additional hypothesis that curvature computation is exact (H_3), the recognition algorithm is successful i.e. *If the Hypotheses H_0 to H_3 are valid, the recognition algorithm classifies correctly all the nodes of a triangulated surface.* Triangles are classified except for those which are based on three frontier nodes. These triangles can be classified in a subsequent model fitting stage. The third hypothesis is restrictive, but numerical experiments showed that the algorithm is succesfull as soon as the surface satisfies hypotheses H_0, H_1 and H_2.

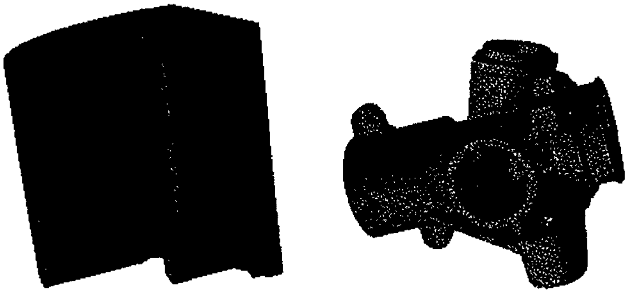

Fig. 3. Two real CAD meshed models.

Table 1 shows the relative number of correctly classified nodes after each step of the algorithm. The first surface satisfies all three hypotheses H_0, H_1 and H_2. The second surface does not satisfy hypotheses H_0 and H_1. As a consequence, some nodes are not classified, and each unclassified node is link to a hypothesis violation.

	example 1	example 2
Initial classification	95 %	99.9 %
Consistency	87 %	80 %
Marching	98 %	88 %
Post treatment	100 %	88 %

Tab. 1. Relative number of classified nodes.

In conclusion, the use of the Hermite approximation scheme we proposed in this paper allows us to build a simple but efficient recognition algorithm. The numerical experiments showed that the Hermite Diffuse Approximation is a powerful tool for surface analysis and partial derivatives estimation. The curvature computation was also used in [11] in a remeshing scheme.

The quality of the curvature estimation on CAD models will help to enlarge the number of surfaces taken into account. Furthermore, [6] showed that Moving Least Square approximation can be applied directly to surface modeling. This approach may be useful for still better curvature estimation.

References

1. Besl, P. J. and R. C. Jain, Three-dimensional object recognition, Computing Surveys **17** (1985), 75–145.
2. Breitkopf, P., G. Touzot, and P. Villon, Explicit form and efficient computation of MLS shape functions and their derivatives, Numerical Method in Engineering, to appear.

3. Duarte, C. A. and J. T. Oden, Hp-clouds, an hp meshless method, Numerical Methods for Partial Differential Equations **16** (1996), 673–705.

4. Guyon, F., Interpolation diffuse. Champs d'atténuation constants, Université de Technologie de Compiègne, 1995.

5. Haralick, R. M. and L. G. Shapiro, *Computer and Robot Vision, Vol 1*, Addison Wesley, New York, 1992.

6. Levin, D., Mesh-independent surface interpolation, Advances in Computational Mathematics, to appear.

7. Liska, T., C. A. M. Duarte, and W. W. Tworzydlo, Hp–meshless cloud method, Computational Methods for Applied Mechanics and Engineering **139** (1996), 3–47.

8. Liska, T. and J. Orkish, The finite difference method at arbitrary irregular grids and its application in applied mechanics, Computers and Structures **11** (1980), 83–95.

9. Mc Ivor, A. M. and P. T. Walkenburg, A comparison of local surface geometry estimation methods, http://www.irl.cri.nz/publications , 1997.

10. McLain, D. H., Drawing contours from arbitrary data points, The Computer Journal **17** (1974), 318–324.

11. Rassineux, A., J. M. Savignat, O. Stab, and P. Villon, Surface remeshing by local Hermite diffuse interpolant, Trends in Unstructured Mesh Generation, USNCCM'99, Boulder, CO.

12. Samson, P. and J. L. Mallet, Curvature analysis of triangulated surfaces in structural geology, Mathematical Geology **29** (1997), 391–412.

13. Savignat, J. M., A. Rassineux, O. Stab, and P. Villon, High resolution curvature analysis and its applications to surface modeling, preprint, 1999.

14. Savignat, J. M., O. Stab, and A. Rassineux, Reconnaissance des surfaces par analyse de la courbure, Tech. report, École des Mines, Paris, 1999.

15. Shi, P., G. Robinson, and J. Duncan, Myocardial motion and function assesment using 4d images, IEEE Conference on Visualization in Biomedical Computing, 1994.

16. Villon, P., Contribution à l'optimisation, Thèse d'état, Univ. de Techn. de Compiègne, 1991.

J. M. Savignat
École des Mines
35, rue Saint Honoré
77 305 Fontainebleau Cedex, France

savignat@cges.ensmp.fr
stab@cges.ensmp.fr
alain.rassineux@utc.fr
pierre.villon@utc.fr